Subject: Re: PostScript and IDL,
Posted by davidf on Tue, 26 May 1998 07:00:00 GMT
View Forum Message <> Reply to Message

Mirko Vukovic (mirko_vukovic@notes.mrc.sony.com) writes:

>> Dick and I have been kicking around the idea of writing a class
>> library of direct graphics objects that we could offer for
>> sale.
>
> No, NO, NOooo!
>
> Not for sale. Give it away freely, let it go forth and multiply,
> and then write books and give lectures for it. You will make some
> money, but will be revered.

This is actually my idea too. I would much rather be revered
than rich. But my wife points out that our retirement program
seems to be carting the kids to all manner of athletic endeavors
in hopes that one of them turns out to be a professional athlete. :-(

My hopes are pinned on the youngest, after he twice scored 6 goals
in a game in a soccer tournament this weekend. :-)

Cheers,

David

David Fanning, Ph.D.
Fanning Software Consulting
E-Mail: davidf@dfanning.com
Phone: 970-221-0438
Coyote's Guide to IDL Programming: http://www.dfanning.com/

Subject: Re: PostScript and IDL,
Posted by mirko_vukovic on Tue, 26 May 1998 07:00:00 GMT
View Forum Message <> Reply to Message

In article <MPG.fd4784e4e6f9f369897ad@news.frii.com>,
davidf@dfanning.com (David Fanning) wrote:

> Dick and I have been kicking around the idea of writing a class
> library of direct graphics objects that we could offer for
> sale.

No, NO, NOooo!

Not for sale. Give it away freely, let it go forth and multiply,
and the write books and give lectures for it. You will make some
money, but will be revered.

(All this coming from a person that has not yet made a cent by
writing or selling programs, nor does depend on that for livelihood).

But even with object graphics, I think the way to think about the whole thing
is as:

-data are objects,

-plots are objects (direct or object)

-files are objects.

You associate data to a plot object and it gets plotted up.

You associate it with a file and it ends up there, or gets here from there.

I think that plot objects should know more then just what they are supposed to
look like, and what the data pairs are. Instead they should know what each
axis is. Thus to transpose the plot axes, I send that to the plot object.

The data object that talks to the plot object does not worry about the
orientation of the x or y axis, it just says this is the x axis data, this is
the y axis data.

the plot object that puts each in its place.

(that kind of implementation got me out of some prorammatic messes in the
past)

cheers,

mirko

Subject: Re: PostScript and IDL,
Posted by mirko_vukovic on Tue, 26 May 1998 07:00:00 GMT
View Forum Message <> Reply to Message

In article <356acbf0.0@wltss01.nerc-wallingford.ac.uk>,
  wmc@bsfiles.nerc-bas.ac.uk wrote:

>
> In article 100000@tesla.drcmr.dk, root <root@tesla.drcmr.dk> writes:
>> Hi, I'm having problems with PostScript generated by IDL_5.1 on a
>> Linux system -- I expect the problem to be general, though.
>>
>> In principle, I'd like to use the Live_tools and other procedures
>> (like Insight) that allows easy printing of plots already on screen.
>>
>> The generated PostScript is, however, rather useless since even the
>> simplest plot generates very large PostScript files. The problem seems
>> to be that the screen buffer is simply copied, rather than redrawing
>> the plot to the printer device. A very large bitmap is therefore
>> written instead of a few PostScript primitives.
>
> According to an IDL course I went on a few weeks ago, this is a feature
> of object graphics & you are stuck with it.
>
> But if you go back to "direct" graphics you should be OK.
>
> - William
>
> ---
> William M Connolley | wmc@bas.ac.uk | http://www.nbs.ac.uk/public/icd/wmc/
> Climate Modeller, British Antarctic Survey | Disclaimer: I speak for myself
>
>
I am ``stuck'' with PCL with the same problem. The way around it is to print
to a file and then send the file to the printer. For some reason, this
process is much faster for me. The funny thing is that the file is called
idl.ps although it is a .pcl file.

mirko

Subject: Re: PostScript and IDL,
Posted by Craig Markwardt on Tue, 26 May 1998 07:00:00 GMT
View Forum Message <> Reply to Message

davidf@dfanning.com (David Fanning) writes:
>
> Dick and I have been kicking around the idea of writing a class
> library of direct graphics objects that we could offer for
> sale. But unfortunately, it appears to be a shrinking
> market. As computers (and, presumably, printers) get faster
> and faster the object graphics advantages begin to

> dominate. It is not at all clear to us which marketplace
> offers the most commercial advantages in the medium term.

One thing to keep in mind is that screen-dumps are rarely suitable for publication. The smooth lines of a postscript figure rendered on a high-resolution printer are usually required for professional journals and other print media.

Craig

--

Craig B. Markwardt, Ph.D. EMAIL: craigmnet@astrog.physics.wisc.edu
Astrophysics, IDL, Finance, Derivatives | Remove "net" for better response

---

Subject: Re: PostScript and IDL,
Posted by wmc on Tue, 26 May 1998 07:00:00 GMT
View Forum Message <> Reply to Message

In article 100000@tesla.drcmr.dk, root <root@tesla.drcmr.dk> writes:
> Hi, I'm having problems with PostScript generated by IDL_5.1 on a
> Linux system -- I expect the problem to be general, though.
>
> In principle, I'd like to use the Live_tools and other procedures
> (like Insight) that allows easy printing of plots already on screen.
>
> The generated PostScript is, however, rather useless since even the
> simplest plot generates very large PostScript files. The problem seems
> to be that the screen buffer is simply copied, rather than redrawing
> the plot to the printer device. A very large bitmap is therefore
> written instead of a few PostScript primitives.

According to an IDL course I went on a few weeks ago, this is a feature of object graphics & you are stuck with it.

But if you go back to "direct" graphics you should be OK.

- William

William M Connolley | wmc@bas.ac.uk | http://www.nbs.ac.uk/public/icd/wmc/
Climate Modeller, British Antarctic Survey | Disclaimer: I speak for myself

---

Subject: Re: PostScript and IDL,

Lars (larsh@magnet.drcmr.dk) writes:

> Hi, I'm having problems with PostScript generated by IDL_5.1 on a
> Linux system -- I expect the problem to be general, though.
>
> In principle, I'd like to use the Live_tools and other procedures
> (like Insight) that allows easy printing of plots already on screen.
>
> The generated PostScript is, however, rather useless since even the
> simplest plot generates very large PostScript files. The problem seems
> to be that the screen buffer is simply copied, rather than redrawing
> the plot to the printer device. A very large bitmap is therefore
> written instead of a few PostScript primitives.

This is correct.

> I wonder, if there is a way to redraw a view generated by a procedure
> like live_plot to a direct graphics device? Other suggestions? Any
> help is appreciated.

There is absolutely no way to draw object graphics in a direct graphics window and visa versa. The two systems are completely separate and distinct. Currently, there is also no way to write object graphics out to a PostScript file in the same format that is used, for example, with direct graphics commands.

There is hope, however. Dick and I have been experimenting with writing LiveTool-like programs that use object programming techniques and direct graphic commands. As I have mentioned here previously, we have been amazed at what is possible. The huge advantage of writing these programs as objects is that they are so incredibly easy to extend. Want another feature? Just add a simple method.

A "plot object" we have recently written for a client has amazing capability. So much so that if I hadn't written it myself I would have a hard time believing that IDL was behind it. Among its many advantages is that it is very fast and it can draw itself in a PostScript file.

You can also take a bit of care in the way you write direct graphics programs so that you can automatically send them to a PostScript file. Most of the second half of my IDL Programming Techniques book explains what these principles are. Many of the programs available on my

web page also illustrate the technique. (See, for example, the XWINDOW program, with which it is fairly easy to build your own "live tools" applications). I am also currently devoting a large portion of my IDL Programming Techniques courses (offered through RSI this year) to learning the advantages of object programming with direct graphics.

Dick and I have been kicking around the idea of writing a class library of direct graphics objects that we could offer for sale. But unfortunately, it appears to be a shrinking market. As computers (and, presumably, printers) get faster and faster the object graphics advantages begin to dominate. It is not at all clear to us which marketplace offers the most commercial advantages in the medium term.

Cheers,

David

David Fanning, Ph.D.
Fanning Software Consulting
E-Mail: davidf@dfanning.com
Phone: 970-221-0438
Coyote's Guide to IDL Programming: http://www.dfanning.com/

Subject: Re: PostScript and IDL,
Posted by steinhh on Wed, 27 May 1998 07:00:00 GMT
View Forum Message <> Reply to Message

I got a very prompt reply from someone at RSI after my previous posting. They are "acutely aware" of most of my concerns, and are looking at different ways to address the problems. They've asked for my comments on a few issues, and I thought I'd give my views here so that other people may express their (dis)agreement.

The bitmap rendering issue has, of course, been driven by complaints about the non-WYSIWYG nature of the 'PS' device compared to the window device, coupled with the big difficulties of mapping object graphics primitives into PS primitives (e.g. pixelized interpolation/textures/stenciling/ zbuffering).

Some questions that were asked by rfrank@rsinc.com:

> How acceptable is non-WYSIWYG printing as

> a trade-off for small, real EPS files?

In one way, the rasterized files ensure *very* consistent rendering. However, wrt. the perceived *quality* of the output when printed on paper, is *not* WYSIWYG with the current implementation. It's more like "one type of WYSIWYG (qualitative) vs another type of WYSIWYG (this *is* what you did see on the screeen)". To me, it's more the perception of the output that counts, rather than the bits & bytes-true-to-life exact copy. So if things are slightly different, but with things appearing in the same place, and conveying the same "general impression" and *qualities* as the window output, I'll be happy. Others may have very different views on this, and my views are certainly colored by an *idea* on what could be achieved, not having been tested on the final trade-off...

The file sizes, however, can be a major obstacle for actually getting to *use* the stuff. Very difficult question.... At least I would like to have a choice in a given situation - i.e., to be able to get *something* on a real-life printer (within a real-life time span :-), but also to be able to get something publishable if I want to, although it may be a post mortem publication due to slow printing :-)

> Would you be willing to break up
> your OG scene into multiple views so that individual views could be
> drawn in raw PS while other views (depending on content) would be
> drawn as bitmaps?

Yes. Definitely. Most certainly. If that's what it takes. This was my initial thought about those OG primitives that *need* to be rasterized in order to get rendered at all. I guess this is the way e.g., shade_surf works in direct graphics...?

But is it necessary for the user to make the distinction between the two, by using individual views? Couldn't it be done at the "atomic" level, automatically?

> If sending out raw PostScript actually generated
> larger files (e.g. PS rasterizing replacement engines could be in the
> file or individual lines/polygons would have to be broken into smaller
> primitives to support correct clipping and composite buffering) would
> that be acceptable?

As far as I can see, this suggestion is really about implementing some kind of OpenGL-type renderer in PostScript -

which *is* an extremely powerful, quite-useable-for-general-purpose-programming language. Of course, I would like to say yes thank you, both! Small files with perfectly rendered 3D graphics objects! But I do understand that such a renderer is a major piece of software - the files could easily end up larger than half a Meg for the renderer alone. And furthermore, they *may* take quite a while to process on a PostScript printer! I once saw a PS program that took less than one page to print (the program itself), but produced a beautiful, silver ball mirroring a chess board stretching out to infinity, with some clouds around it. It took on the order of a day to process :-)

Again, I'd suggest that the user be given a *choice* between various output modes. Perhaps an adjustable-resolution bitmap rendering mode (which could be used as a "draft" mode with low resolution, producing manageable files, or as a "publication" mode with high resolution for those with enough disk space and printer capabilities), plus a PostScript software rendering mode, that may produce *large* files for those graphics hierarchies that need to have hi-tech rendering. (But remember that it's possible to make subsets of the necessary preamble - you don't need a huge rendering library to do a line plot conversion from OG -> PS primitives!).

It seems like the implementation of an adjustable-resolution bitmap rendering mode should be a piece of cake, given what's already there... But I'd like to have at least some mode that's not rasterizing my line plots/wire meshs etc..

RSI are continually trying to judge what compromizes to make based on input from us - so please make your voice heard!

Regards,

Stein Vidar

Subject: Re: PostScript and IDL,
Posted by steinhh on Wed, 27 May 1998 07:00:00 GMT
View Forum Message <> Reply to Message

In my opinion, the PostScript output from object graphics is so lame (both in quality and file size) that it is endangering the whole idea of using object graphics for anything that you may at some point in time want to publish on paper.

This is a bit of ranting over some details that irritate me enormously. For some reason, it seems that the OG printer output interface was written by someone with experience exclusively from PC's, with a very cheap (low-resolution) printer hanging on the side, hardwired into the motherboard for instantaneous input of whatever is on the screen in a WYSIWYG manner :-). Never any *real* reason to create *real* encapsulated postscript files with any controllable size or anything!

With the normal 'PS' device you can control the sizes etc. of the EPS output, but no, this object graphics stuff is so great it always deserves a full page by itself..? With a showpage command thrown into the file for good measure, so the whole thing will print if you send it straight to the printer in stead of encapsulating it in a real document (just points out the mindset of whoever wrote the stuff). This may cause problems for some programs that try to manipulate/overplot stuff on the EPS file.

Even the simplest line plot (Generic, EPSF format) comes out to about half a MB. Imagine writing a thesis with a hundred of those - since you need to have the original file as well as one copy included in your PS file, you spend 100MB on plots!

If you throw the thing at the printer, the lines are clearly more jagged than direct graphics output. The lines are thinner, apparently just one "pixel" wide, and if you try to shrink the full page to fit into a document, the subsampling of the pixelized image seems to throw out pixel rows/lines, making everything in your plot look like it's produced with a dash-dash linestyle.

And if I ever write a *program* that produces postscript output from object graphics, I would like to have the ability to have some control over where the files end up, without having to interrogate the user (myself or whoever) every bloody time I want to send stuff to a new file! Please, a printer->setproperty,filename=<string>!

I do like IDL's object graphics, conceptually it's very neat and *extremely* powerful - though you *really* need to have good graphics acceleration harware! (I don't have it yet).

But unless someone provides a method to produce sensible

postscript output, I for one will be very reluctant about using object graphics at all.

Although the internal bitmap rendering process of OG PS files ensures "consistent" looks between screen/postscript output, it is simply *no* good for producing quality output for publication! We need better resolution for lines *and* smaller files!

Given an OG hierarchy, most of the objects contents can be rendered quite faithfully in direct graphics without using pixelization (shaded surfaces/texture mapped surfaces/lighted surfaces etc excluded). Maybe someone could write a direct graphics "renderer" of OG hierarchies? Please..?

Or maybe what we should hope for is an OpenGL printer standard...?

Regards,

Stein Vidar

Subject: Re: PostScript and IDL,
Posted by davidf on Thu, 28 May 1998 07:00:00 GMT
View Forum Message <> Reply to Message

Hi Folks,

It seems this discussion has pricked a number of sympathetic ears at RSI. But the developers who have to make the decisions need help from us as to what we really want. Here is how one developer framed the problem for me:

> The big issue for us is what tradeoffs are users willing
>
> will have to give up speed/memory/filesize or WYSIWYG to get the type of PS they are looking for.
>
> What sort of rendering would you be willing to give up to get
>
> individual views are rendered as bitmaps or raw PS depending
>
> about a device which rendered only what is possible with

PS (i.e. some objects would disappear completely)?

These are the real issues we are struggling with right now...
Any input you have would be helpful.

Immediately after a new release is when the big discussions
about what to do next go on. There is a lot of horse trading
between marketing and the developers over what is needed and
what is possible in the given time frame. This is absolutely
the time when users can have the most influence over what happens
next. If you have ideas about this or anything else you like/dislike
about IDL, this would be a good time to get those fingers
working. I should think anyone at support@rsinc.com would
be happy to pass your comments along to the appropriate
people.

Cheers,

David

David Fanning, Ph.D.
Fanning Software Consulting
E-Mail: davidf@dfanning.com
Phone: 970-221-0438
Coyote's Guide to IDL Programming: http://www.dfanning.com/